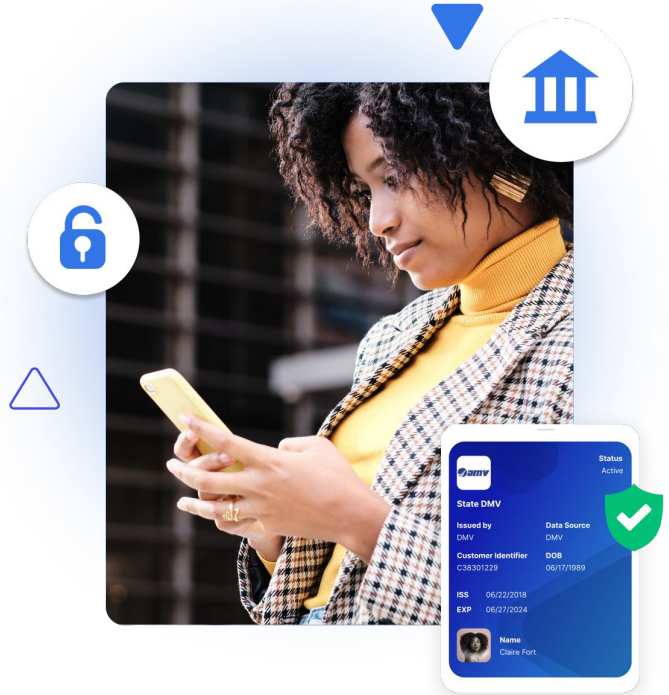
# SD-JWT-based Verifiable Credentials (SD-JWT VC)

draft-ietf-oauth-sd-jwt-vc-01

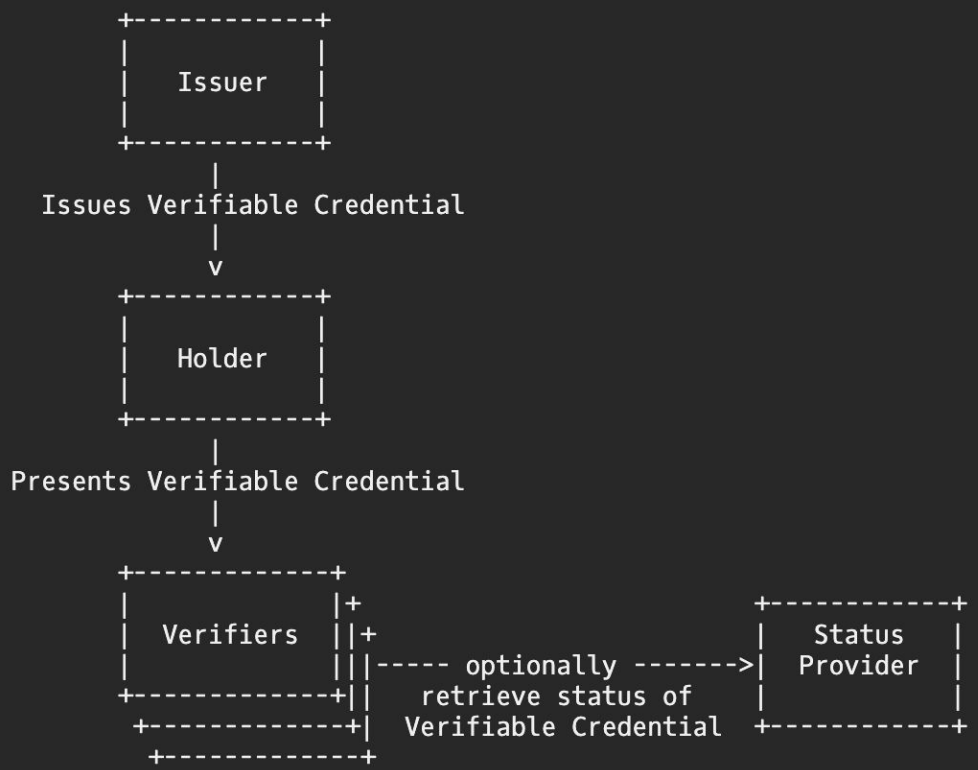
Oliver Terbu
Daniel Fett (Authlete)

# Recap: SD-JWT Verifiable Credentials (SD-JWT VC)

- Defines a profile of SD-JWT for Verifiable Credentials, similar to an ID Token is a specific profile of JWT
- Defines data formats and media types for Verifiable Credentials based on SD-JWTs with JSON payloads
- Defines validation and processing rules for verifiers and holders of Verifiable Credentials based on SD-JWTs
- Includes support for plain JWTs (non-SD)



# Issuer-Holder-Verifier model



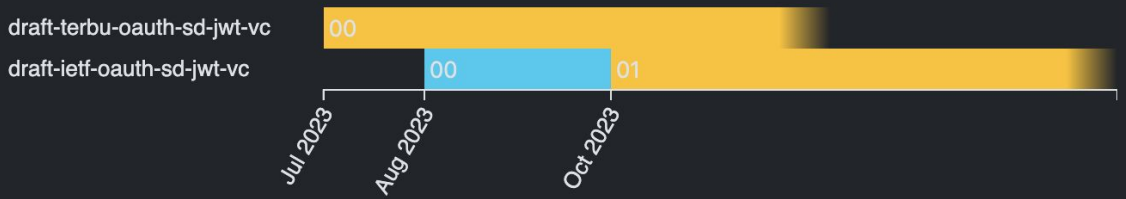
# SD-JWT-based Verifiable Credentials (SD-JWT VC)

## draft-ietf-oauth-sd-jwt-vc-01

Status | IESG evaluation record | IESG writeups | Email expansions | History

**Versions:**

00 01




| | | |
|---|---|---|
| **Document** | **Type** | Active Internet-Draft (oauth WG) |
| | **Authors** | Oliver Terbu ✉, Daniel Fett ✉ |
| | **Last updated** | 2023-10-23 (Latest revision 2023-08-16) |
| | **Replaces** | draft-terbu-oauth-sd-jwt-vc |
| | **RFC stream** | Internet Engineering Task Force (IETF) |

# What’s new?

# What’s new in version -01?

- Type identifier
  - New rules for collision-resistant names
  - Renamed `type` JWT claim to `vct`
- Removed duplicated and inconsistent requirements on KB-JWT
- Defined rules for obtaining issuer verification key
- Editorial changes and fixed references

# Obtaining issuer verification key

- If `x5*` JWT headers present in Issuer-signed JWT,
  - and if `iss` value matches `dNSName` (using DNS URI scheme) or `uniformResourceIdentifier` in SAN extension.
  - Then, get verification key from X.509 leaf certificate
- If `https://example.com`.
  - Then, get verification key from .well-known JWT Issuer Metadata
- If `did:example:123456`.
  - Then, get verification key from DID Resolution result,
  - and optionally use `kid` JWT claim to further locate the key in the DID Document via absolute/relative DID URL
- Ecosystems can define their own rules as long as they don't contradict the rules above.

# Discussion and Next Steps

# What is a Credential Type?

There is a set of **metadata** usually associated with each S**D-JWT VC type:**

- Display information for the credential
- Information about the claims, including
  - Display information
  - Type information
  - Status information (self-attested vs. verified etc.)
- The binding supported/usually used for this credential type

This information helps **Wallets**, **Verifiers,** and **Developers**

# Defining the Concept “SD-JWT VC Credential Type”

We should define what SD-JWT VC Credential Type means!

- Define set of data
- Define method for distribution
  - For registered types: in the registry
  - For types that are URLs: discoverable via a .well-known URL
- Only for SD-JWT VC — *not* format-agnostic
- Check what we can remove from OpenID4VCI and the High-Assurance Profile

# Completely Made-Up Example

```
{
    "vct": "https://credentials.example.com/example1",
    "display": [ // display information for the credential
        {
            "en-US": {
                "name": "University Credential",
                "logo": {
                    "url": "https://exampleuniversity.com/public/logo.png",
                    "alt_text": "a square logo of a university"
                },
                "background_color": "#12107c",
                "text_color": "#FFFFFF"
            }
        }
    ],
    "claims": // mapping from JSON pointer to information about this claim
    {
        "/degree": {
            "display": {
                "de-DE": {
                    "label": "Abschluss",
                    "description": "Der Abschluss des Studenten"
                },
                "en-US": {
                    "label": "Degree",
                    "description": "Degree earned by the student"
                }
            },
            "status": "verified"
        },
        // ...
    }
    // ...
}
```

# Renaming JWT Issuer Metadata

- Proposals
  - SD-JWT VC Issuer Metadata
  - VC Issuer Metadata
  - ...

# What else?

- eIDAS 2.0 ARF examples
- Confidence and assurance levels, and binding methods

# Thank you!